```
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
from scipy import stats
```

```
nrgvsecon=pd.read_csv("africanrgvsgdrp.csv",index_col=[0])
nrgvsecon.shape
```

(52, 12)

Electricity Generation Sources in each country

```
nrgvsecon.head(10)
```

	country	population	real gdp per capita $	installed capacity kW	fossil fuels	nuclear	solar	wind	hydroelectricity	tide and wave	geothermal	bic
0	nigeria	225082083.0	4900.0	11691000.0	78.1	0.0	0.2	0.0	21.7	0.0	0.0	
1	egypt	107770524.0	12000.0	59826000.0	88.7	0.0	1.0	2.5	7.7	0.0	0.0	
2	south-africa	57516665.0	11500.0	62728000.0	87.9	5.2	1.6	2.6	2.5	0.0	0.0	
3	algeria	44178884.0	10700.0	21694000.0	98.9	0.0	0.9	0.0	0.1	0.0	0.0	
4	morocco	36738229.0	6900.0	14187000.0	81.6	0.0	1.1	13.0	4.4	0.0	0.0	
5	angola	34795287.0	6200.0	7344000.0	28.4	0.0	0.1	0.0	70.1	0.0	0.0	
6	kenya	55864655.0	4200.0	3304000.0	8.3	0.0	1.0	10.7	32.6	0.0	46.2	
7	ethiopia	113656596.0	2300.0	4856000.0	0.0	0.0	0.1	3.8	95.8	0.0	0.0	
8	tanzania	63852892.0	2600.0	1623000.0	65.0	0.0	1.3	0.0	32.8	0.0	0.0	
9	ghana	33107275.0	5300.0	5312000.0	63.8	0.0	0.3	0.0	35.9	0.0	0.0	

```
import matplotlib as mpl

mpl.rcParams['axes.spines.top'] = False
mpl.rcParams['axes.spines.right'] = False
```

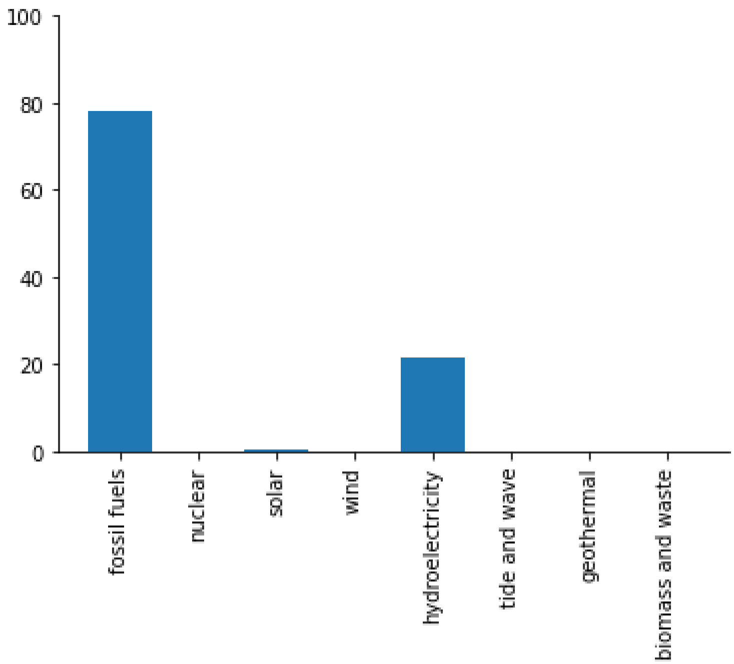
```
nrgvsecon.iloc[0]
```

```
country                    nigeria
population             225082083.0
real gdp per capita $       4900.0
installed capacity kW   11691000.0
fossil fuels                  78.1
nuclear                        0.0
solar                          0.2
wind                           0.0
hydroelectricity              21.7
tide and wave                  0.0
```

```
geothermal                     0.0
biomass and waste              0.1
Name: 0, dtype: object
```

```
x=nrgvsecon.iloc[0].index[4:]
y=nrgvsecon.iloc[0].values[4:]
orderedy=nrgvsecon.iloc[0].value_counts().index
fig=plt.bar(x,y)
plt.xticks(rotation=90)
plt.ylim(0,100)


plt.show()
```



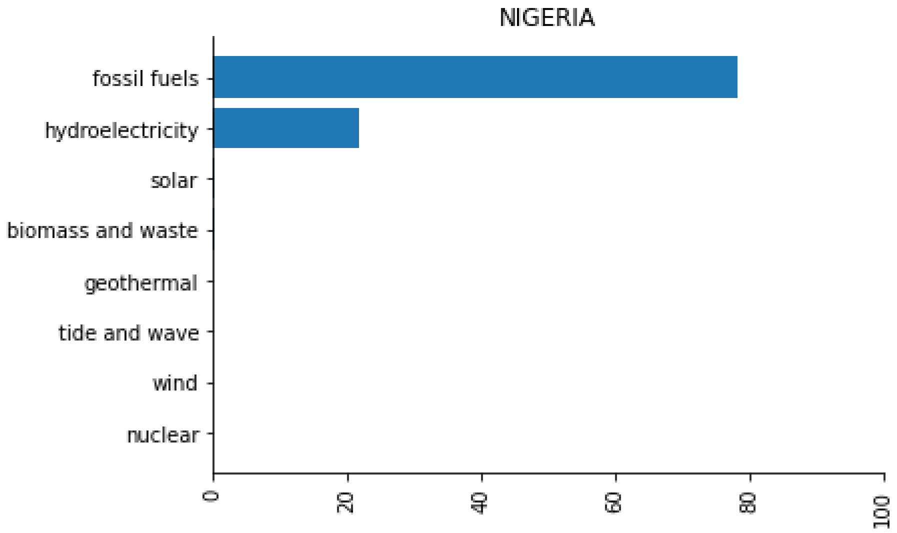
We can see the comparisons of each electricity generation source in a country(in this case, Nigeria).

Now, a better way to visualize this would be the bar charts in some order.e.g. the highest sources to the lowest.

```
# argsort returns an array of indices in order ascending order
order=np.argsort(nrgvsecon.iloc[0].values[4:])
```

```
# barh will plot a horizontal bar instead of a vertical one
x=nrgvsecon.iloc[0].index[4:][order]
y=nrgvsecon.iloc[0].values[4:][order]
orderedy=nrgvsecon.iloc[0].value_counts().index
fig=plt.barh(x,y)
plt.xticks(rotation=90)
plt.xlim(0,100)
plt.title(nrgvsecon['country'][0].upper())

plt.show()
```

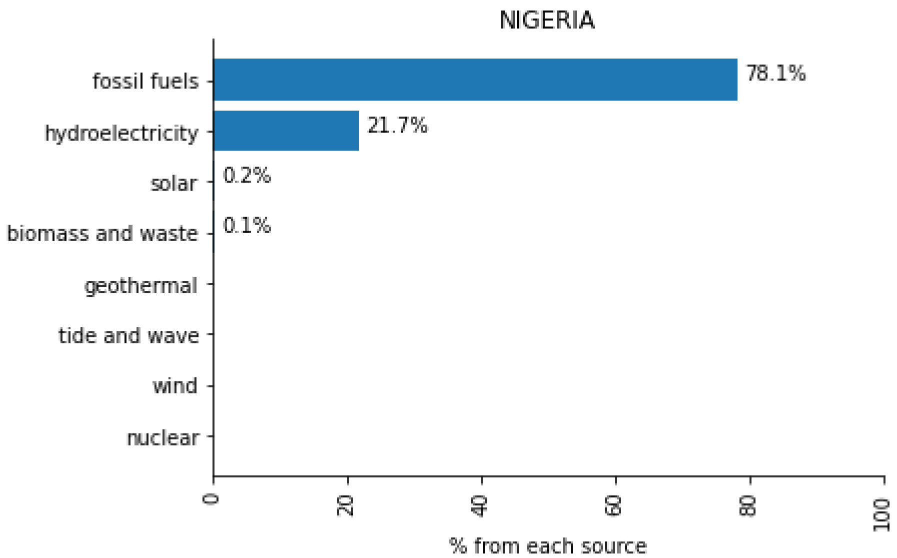


Another interesting and more clear way to do things would be to have the percent appear at the end of each bar.

We can use plt.text to do this

```
x=nrgvsecon.iloc[0].index[4:][order]
y=nrgvsecon.iloc[0].values[4:][order]
orderedy=nrgvsecon.iloc[0].value_counts().index
fig=plt.barh(x,y)
plt.xticks(rotation=90)
plt.xlim(0,100)
plt.xlabel("% from each source")
plt.title(nrgvsecon['country'][0].upper())
for i in range(len(y)):
    if y[i]!=0:
        plt.text(y[i]+1, i,f"{y[i]}%")

plt.show()
```




```
# test
# nrgvsecon[nrgvsecon['country'].str.contains('kenya')].iloc[0][2:].index
```

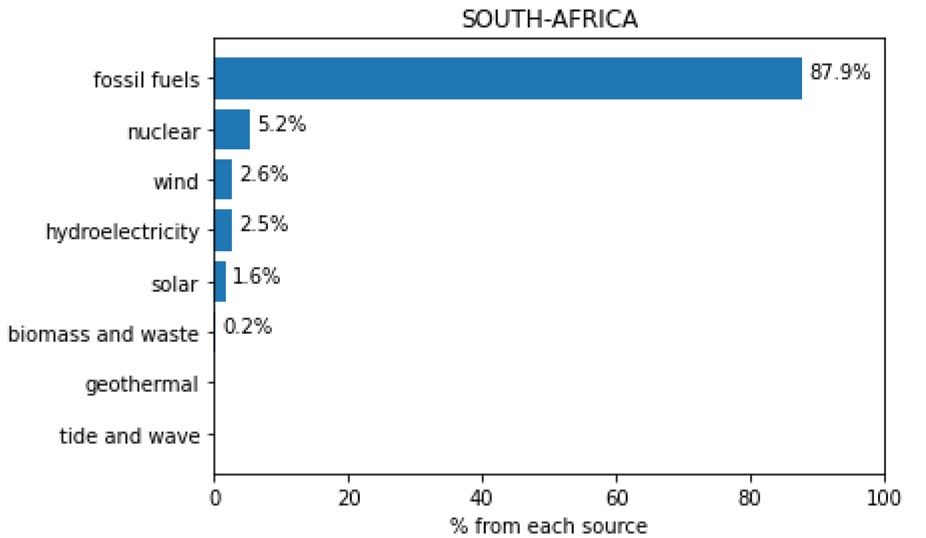
The information on this bar chart is really clear.

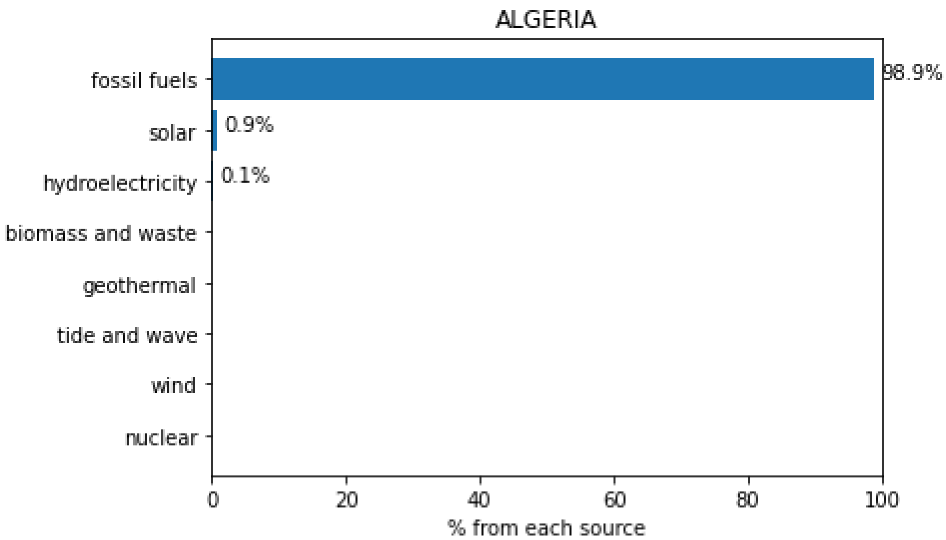
We can see what the percentage contribution from each electricity generation source is!

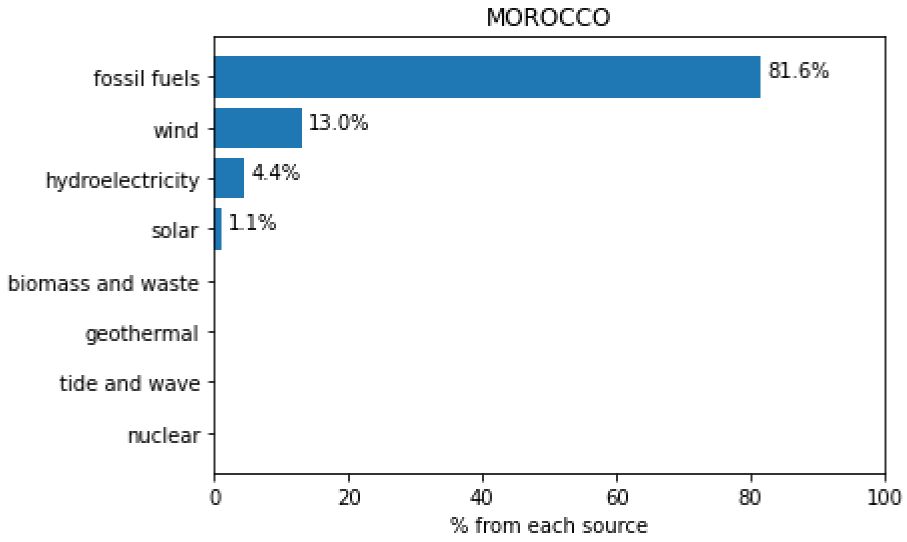
Let's create a function to get this information from any country.

```
def plotpercountry(countries):
    for country in countries:
        order=np.argsort(nrgvsecon[nrgvsecon['country'].str.contains(country)].iloc[0].
        y=nrgvsecon[nrgvsecon['country'].str.contains(country)].values[0][4:][order]
        x=nrgvsecon[nrgvsecon['country'].str.contains(country)].iloc[0][4:].index[order
        plt.barh(x,y)
        plt.xlim(0,100)
        plt.xlabel("% from each source")
        plt.title(country.upper())
        for i in range(len(y)):
            if y[i]!=0:
                plt.text(y[i]+1, i,f"{y[i]}%")
        plt.show()
```

```
plotpercountry(nrgvsecon.country.values[2:5])
```








```
nrgvsecon.head(10)
```

	country	population	real gdp per capita $	installed capacity kW	fossil fuels	nuclear	solar	wind	hydroelectricity	tide and wave	geothermal	bic
0	nigeria	225082083.0	4900.0	11691000.0	78.1	0.0	0.2	0.0	21.7	0.0	0.0	
1	egypt	107770524.0	12000.0	59826000.0	88.7	0.0	1.0	2.5	7.7	0.0	0.0	
2	south-africa	57516665.0	11500.0	62728000.0	87.9	5.2	1.6	2.6	2.5	0.0	0.0	
3	algeria	44178884.0	10700.0	21694000.0	98.9	0.0	0.9	0.0	0.1	0.0	0.0	
4	morocco	36738229.0	6900.0	14187000.0	81.6	0.0	1.1	13.0	4.4	0.0	0.0	
5	angola	34795287.0	6200.0	7344000.0	28.4	0.0	0.1	0.0	70.1	0.0	0.0	
6	kenya	55864655.0	4200.0	3304000.0	8.3	0.0	1.0	10.7	32.6	0.0	46.2	
7	ethiopia	113656596.0	2300.0	4856000.0	0.0	0.0	0.1	3.8	95.8	0.0	0.0	
8	tanzania	63852892.0	2600.0	1623000.0	65.0	0.0	1.3	0.0	32.8	0.0	0.0	
9	ghana	33107275.0	5300.0	5312000.0	63.8	0.0	0.3	0.0	35.9	0.0	0.0	

Comparing energy sources

```
nrgvsecon.sum(numeric_only=True)
```

```
population                1.394572e+09
real gdp per capita $     2.809000e+05
installed capacity kW     2.433790e+08
fossil fuels              3.099700e+03
nuclear                   5.200000e+00
solar                     7.930000e+01
wind                      6.640000e+01
hydroelectricity          1.825700e+03
tide and wave             0.000000e+00
geothermal                4.620000e+01
biomass and waste         7.870000e+01
dtype: float64
```

```
totalcapacity=nrgvsecon['installed capacity kW'].sum()
```

```
(np.sum(nrgvsecon['installed capacity kW']*nrgvsecon['fossil fuels']/100))/(nrgvsecon['
```

```
0.7685277612283722
```

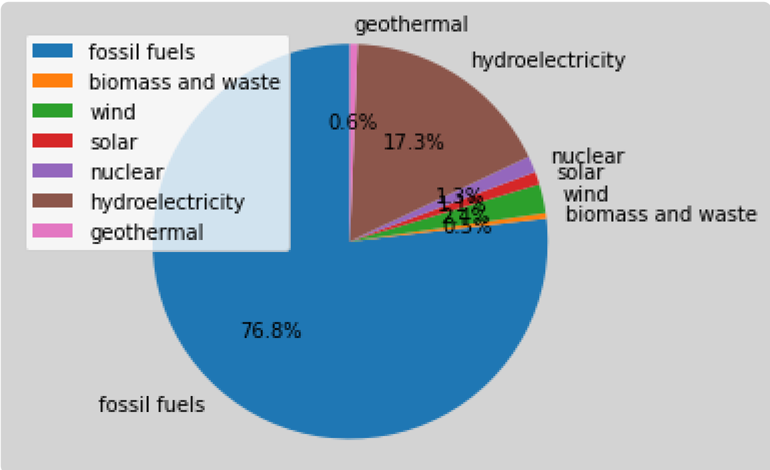
```
sources=['fossil fuels','biomass and waste','wind','solar','nuclear','hydroelectricity'
```

```
def sumcapacity(sources):
  capacities=[]
  for source in sources:
    capacities.append((np.sum(nrgvsecon['installed capacity kW']*nrgvsecon[source]/100)
  return capacities
```

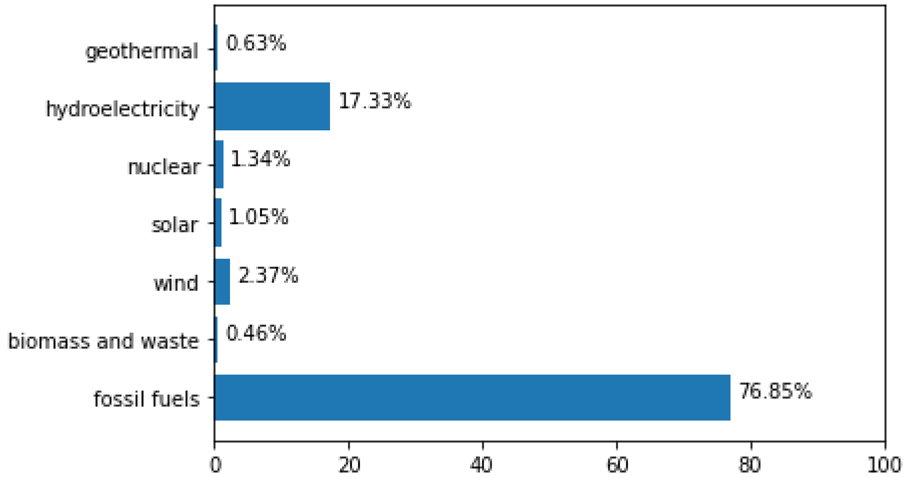
```
capas=(sumcapacity(sources))
```

```
fig1, ax1 = plt.subplots()
ax1.pie(capas, labels=sources, autopct='%1.1f%%',
        startangle=90)
ax1.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
fig1.set_facecolor('lightgrey')
ax1.legend()

plt.show()
```




```
mpl.rcParams['axes.spines.top'] = True
mpl.rcParams['axes.spines.right'] = True
plt.barh(sources,(np.array(capas)*100))
plt.xlim(0,100)
for i in range(len(capas)):
    if capas[i]!=0:
        plt.text(capas[i]*100+1, i,f"{(capas[i]*100):.2f}%")
plt.show()
```



gdp

```
nrgvsecon.head()
```

	country	population	real gdp per capita $	installed capacity kW	fossil fuels	nuclear	solar	wind	hydroelectricity	tide and wave	geothermal	bic
0	nigeria	225082083.0	4900.0	11691000.0	78.1	0.0	0.2	0.0	21.7	0.0	0.0	
1	egypt	107770524.0	12000.0	59826000.0	88.7	0.0	1.0	2.5	7.7	0.0	0.0	
2	south-africa	57516665.0	11500.0	62728000.0	87.9	5.2	1.6	2.6	2.5	0.0	0.0	
3	algeria	44178884.0	10700.0	21694000.0	98.9	0.0	0.9	0.0	0.1	0.0	0.0	
4	morocco	36738229.0	6900.0	14187000.0	81.6	0.0	1.1	13.0	4.4	0.0	0.0	

```
np.log10(nrgvsecon['real gdp per capita $'].describe())
```

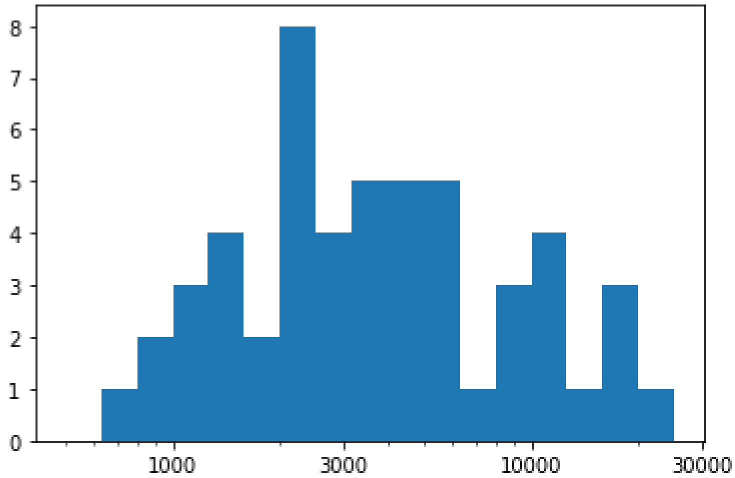
```
count    1.716003
mean     3.732548
std      3.720144
min      2.845098
25%      3.322219
50%      3.518514
75%      3.804480
max      4.387390
Name: real gdp per capita $, dtype: float64
```

```
10**np.arange(2.7,4.4+0.1,0.1)
```

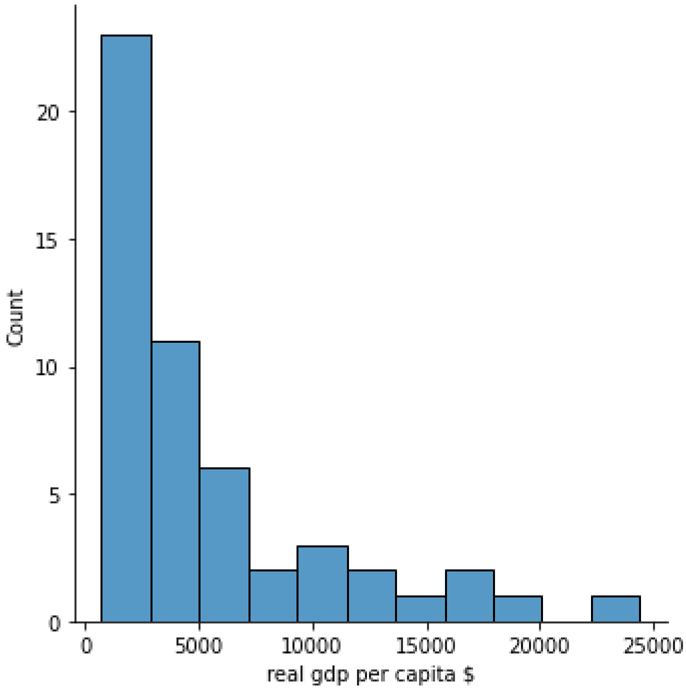
```
array([  501.18723363,   630.95734448,   794.32823472,  1000.        ,
        1258.92541179,  1584.89319246,  1995.26231497,  2511.88643151,
        3162.27766017,  3981.07170553,  5011.87233627,  6309.5734448 ,
        7943.28234724, 10000.        , 12589.25411794, 15848.93192461,
       19952.62314969, 25118.8643151 ])
```

```
bins=10**np.arange(2.7,4.4+0.1,0.1)
ticks=[1000,3000,10000,30000]
```

```
labels=[f'{v}' for v in ticks]
plt.hist(data = nrgvsecon, x='real gdp per capita $',bins=bins)
plt.xscale('log');
plt.xticks(ticks,labels);
```




```
sns.displot(nrgvsecon['real gdp per capita $']);
```




```
nrgvsecon.sort_values('installed capacity kW')
```

```
stats.spearmanr(nrgvsecon['installed capacity kW'],nrgvsecon['real gdp per capita $'])
```

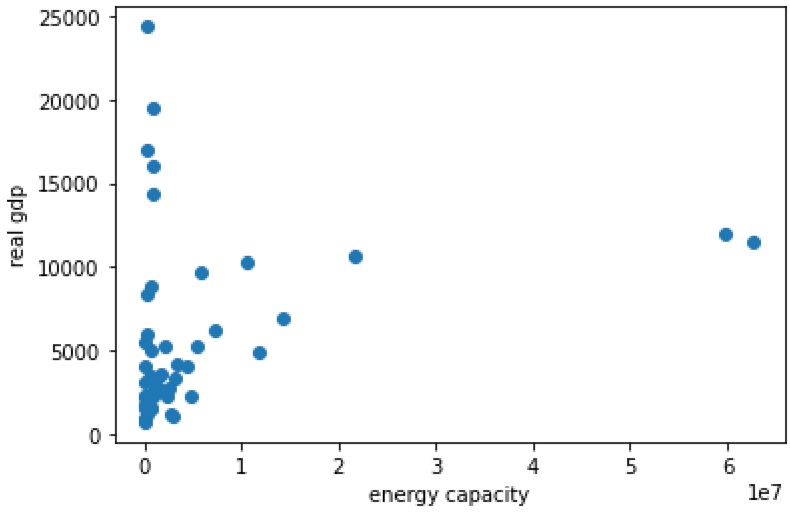
```
SpearmanrResult(correlation=0.42678686304592384, pvalue=0.0016038790372016702)
```

```
x=nrgvsecon['installed capacity kW']
y=nrgvsecon['real gdp per capita $']

plt.xlabel('energy capacity')
```

```
plt.ylabel('real gdp')
plt.scatter(x,y)
```

```
<matplotlib.collections.PathCollection at 0x2bc01de83a0>
```




```
installed=nrgvsecon['installed capacity kW']
```

```
gdp=nrgvsecon['real gdp per capita $']
```

```
instmean=np.mean(nrgvsecon['installed capacity kW'])
instmean
```

```
4680365.384615385
```

```
gdpmean=np.mean(nrgvsecon['real gdp per capita $'])
```

```
abovemean=([item for item in installed if item>instmean])
```

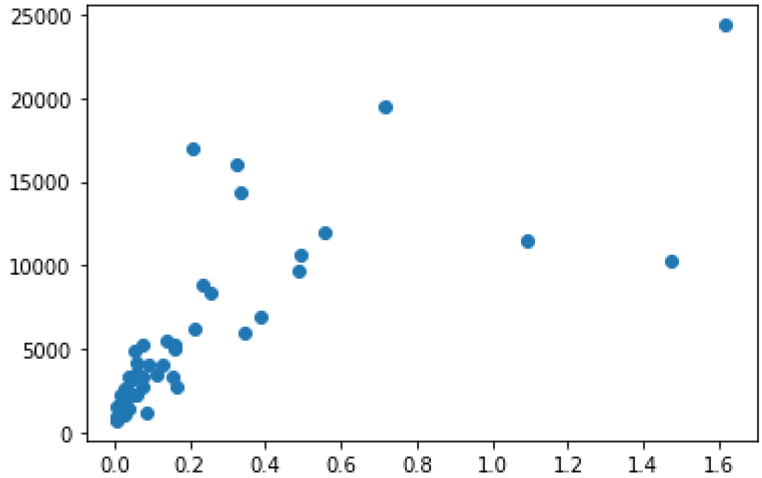
Taking population to account

```
prod=nrgvsecon['installed capacity kW']/nrgvsecon['population']
```

```
gdp=nrgvsecon['real gdp per capita $']
```

```
plt.scatter(prod,gdp)
```

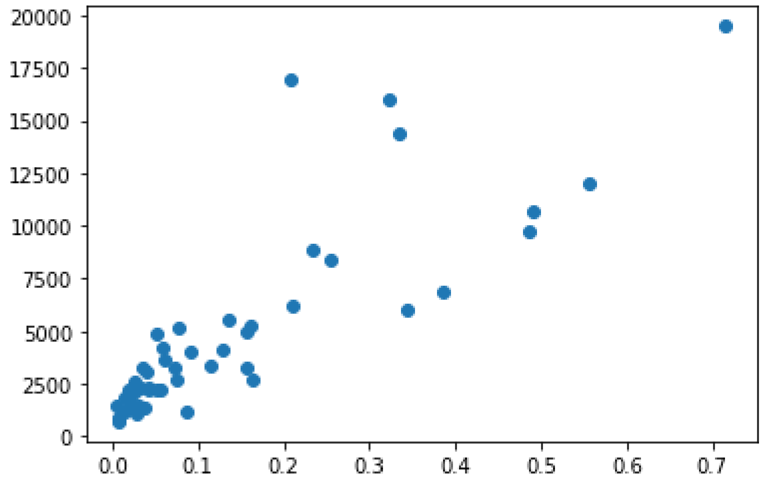
```
<matplotlib.collections.PathCollection at 0x2bc01e2b7f0>
```




```
prodind=np.argsort(prod)
```

```
plt.scatter(prod[prodind][:49],gdp[prodind][:49])
```

```
<matplotlib.collections.PathCollection at 0x2bc01e85f00>
```




```
gdp[prodind][:49]
```

```
from scipy import stats
```

```
stats.spearmanr(prod,gdp)
```

```
SpearmanrResult(correlation=0.9038651435156675, pvalue=4.521208510127803e-20)
```

```
stats.pearsonr(prod,gdp)
```

```
PearsonRResult(statistic=0.7709646191018527, pvalue=2.293666649612281e-11)
```

```
import jovian
```

```
jovian.commit(filename="exploreafnrgvsecon.ipynb")
```

```
[jovian] Updating notebook "andrewkamaukim/exploreafnrgvsecon" on https://jovian.ai/
```

```
[jovian] Committed successfully! https://jovian.ai/andrewkamaukim/exploreafnrgvsecon

'https://jovian.ai/andrewkamaukim/exploreafnrgvsecon'
```